CLAIMS:

1. A method of scheduling a plurality of tasks in a data processing system, comprising the steps of:

defining each task of said plurality such that a synchronization primitive releasing resources that matches another synchronization primitive protecting resources contained therein does not span a task boundary;

specifying a subset of tasks as preemptible or as non-preemptible depending on whether or not the tasks protect usage of at least one same resource;

for each task of the plurality, providing suspension data specifying suspension of the task based on memory used thereby and on the specified preemptability of the task;

processing one of the plurality of tasks;

monitoring for an input indicative of memory used by the task matching the suspension data associated with the task; and

if said suspension data specifies said task is preemptible, performing the steps of:

(i) suspending said task on the basis of said monitored input,

(ii) executing synchronization primitives with respect to the protected resources of the suspended task until said suspended task terminates, and

(ii) processing a different one of the plurality.

2. The method of claim 1, wherein said input comprises data indicative of a suspension request.

3. The method of claim 2, further comprising the steps of:

receiving first data identifying maximum memory usage associated with the plurality of tasks;

receiving second data identifying memory available for processing the plurality of tasks; and

identifying, on the basis of the first and second data, whether there is sufficient memory available to process the tasks;

wherein, said monitoring, suspending steps, and executing steps are performed only in response to identifying insufficient memory.

4. The method of claim 3, further comprising the steps of:

monitoring termination of tasks; and

in response to a task termination, repeating said step of identifying availability of memory in response to a task terminating.

5. The method of claim 4, in which, in response to identifying sufficient memory to execute the remaining tasks, the monitoring step is deemed unnecessary.

6. The method of claim 1, further comprising the steps of:

receiving first data identifying maximum memory usage associated with the plurality of tasks;

receiving second data identifying memory available for processing the plurality of tasks; and

identifying, on the basis of the first and second data, whether there is sufficient memory available to process the tasks;

wherein, said monitoring, suspending, and executing steps are performed only in response to identifying insufficient memory.

7. The method of claim 6, further comprising the steps of:

monitoring termination of tasks;

in response to a task termination, repeating said step of identifying availability of memory.

8. The method of claim 7, in which, in response to identifying sufficient memory to execute the remaining tasks, the monitoring step is deemed unnecessary.

9. The method of claim 1, further comprising the steps of:

receiving first data identifying maximum memory usage associated with the plurality of tasks;

receiving second data identifying memory available for processing the plurality of tasks; and

identifying, on the basis of the first and second data, whether there is sufficient memory available to process the tasks;

wherein, said monitoring, suspending, and executing steps are performed only in response to identifying insufficient memory.

10. The method claim 9, further comprising the steps of:

monitoring termination of tasks; and

in response to a task termination, repeating said step of identifying availability of memory.

11. The method according to claim 10, in which, in response to identifying sufficient memory to execute the remaining tasks, the monitoring step is deemed unnecessary.

12. A scheduler for use in a data processing system, the data processing system being arranged to execute a plurality of tasks defined such that a synchronization primitive releasing resources matching another synchronization primitive protecting resources contained therein does not span a task boundary and having access to a specified amount of memory for use in executing the tasks, the scheduler comprising:

a data receiver arranged to receive data identifying maximum memory usage associated with a task, exclusive resource usage of the task, and preemptability of the task, wherein a subset of said plurality of tasks protecting usage of the same resource are all identified as one of preemptible or non-nonpreemptible;

an evaluator arranged to identify, on the basis of the received data, whether there is sufficient memory to execute the tasks; and a selector arranged to select at least one task for suspension during execution of the task, said suspension coinciding with a specified memory usage by the task and the task being preemptible;

wherein, in response to the evaluator identifying that there is insufficient memory to execute the plurality of tasks,

- the selector selects at least one task for suspension, on the basis of its specified memory usage and its preemptability, and the specified amount of memory available to the data processing system,
- the scheduler suspends execution of the at least one selected task in response to the task using the specified memory and the task being preemptible, and
- the evaluator directs execution thereafter of synchronization primitives with respect to the protected resources of the suspended at least one task. until said suspended at least one task terminates.

13. A scheduler according to claim 12, wherein the evaluator is further arranged to monitor termination of tasks, and in response to a task terminating, to identify whether there is sufficient memory to execute the remaining tasks.

14. A scheduler according to claim 13, wherein in response to the evaluator identifying sufficient memory to execute the remaining tasks the selector is arranged to deselect said selected at least one task.

15. A data processing system arranged to execute a plurality of tasks having each task of said plurality defined such that a synchronization primitive matching another synchronization primitive contained therein does not span a task boundary, the data processing system including:

memory arranged to hold instructions and data during execution of a task;

receiving means arranged to receive data identifying maximum memory usage associated with a task and data specifying preemptability of the task;

evaluating means arranged to identify, on the basis of the received data, whether there is sufficient memory to execute the tasks and whether the tasks are preemptible; and

a scheduler arranged to schedule execution of the tasks on the basis of input received from the evaluating means,

wherein, in response to identification of insufficient memory to execute the plurality of tasks,

the scheduler is arranged to suspend execution of at least one task in dependence on memory usage by the task, exclusive resource usage by the task, and preemptabilty of the task and to direct the execution thereafter of synchronization primitives with respect to the protected resources of the suspended at least one task until said suspended task terminates.

16. The data processing system of claim 15, wherein a subset of said plurality of tasks is determined be preemptible or non-preemptible depending on whether or not the subset of tasks protect usage of the same resource.

17. A method of transmitting data to a data processing system, the method comprising:

defining a task such that a synchronization primitive that protects usage of a resource that matches another synchronization primitive contained therein does not span the task boundary;

defining all tasks as preemptible or as non-preemptible depending on whether or not the tasks protect usage of at least one same resource;

transmitting data for use by the data processing system in processing the task; and

transmitting suspension data specifying suspension of the task based on memory usage and preemptability during processing thereof,

wherein the data processing system is configured to perform a process comprising:

monitoring for an input indicative of memory usage of the task matching the suspension data associated with the task; and

if said suspension data specifies the task is preemptible, suspending processing of said task on the basis of said monitored input and thereafter executing synchronization primitives with respect to the resources protected by the suspended task until the suspended task terminates.

18. A method according to claim 17, wherein the suspension data includes data identifying maximum memory usage associated with the task, exclusive resource usage associated with the task, and preemptability of the task.

19. A method according to claim 17, wherein the suspension data identifies at least one point at which processing of the task can be suspended, based on memory usage of the task, exclusive resource usage of the task, and preemptability of the task.

20. A method according to claim 19, wherein the task comprises a plurality of sub-jobs and said data identifying at least one point at which processing of the task can be suspended corresponds to each such sub-job that is preemptible.

21. A method according to claim 19, wherein the suspension data includes data identifying maximum memory usage associated with the task and exclusive resource usage associated with the task.

22. A method according to claim 21, wherein the task comprises a plurality of sub-jobs and said identifying at least one point at which processing of the task can be suspended corresponds to each such sub-job that is preemptible.

23. A method of configuring a task for use in a data processing system, the method including associating suspension data with the task, the suspension data specifying suspension of the task based on memory usage associated therewith, exclusive resource usage of the task, and preemptability of the task, wherein the data processing system is arranged to perform a process in respect of a plurality of tasks, the process comprising:

defining the task such that a synchronization primitive matching another synchronization primitive contain there does not span a task boundary;

monitoring for an input indicative of memory usage of the task matching the suspension data associated with the task; and

if the suspension data specifies said task is preemptible,

- suspending processing of said task on the basis of said monitored input, and
- executing thereafter synchronization primitives with respect to the exclusively used resources of the suspended at least one task until said task terminates.

24. A computer program stored in a memory, comprising a set of instructions arranged to cause a processing system to perform the method according to of claim 1.